UNITED STATES PATENT AND TRADEMARK OFFICE

A

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/239,194	01/28/1999	JOHN S YATES JR.	114596-05-4013	9716

38492 7590 09/30/2005

WILLKIE FARR & GALLAGHER LLP
INTELLECTUAL PROPERTY LEGAL ASSISTANTS
787 SEVENTH AVE
NEW YORK, NY 10019-6099

EXAMINER	
TANG, KENNETH	

ART UNIT	PAPER NUMBER
2195	

DATE MAILED: 09/30/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)
	09/239,194	YATES ET AL.
	Examiner	Art Unit
	Kenneth Tang	2195

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *18 July 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-83* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-83* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *7/18/05*.
4) ☒ Interview Summary (PTO-413) Paper No(s)/Mail Date. *9/7/05*.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the Request for Pre-Appeal Conference filed on 7/18/05.

2. In the Amendment/Remarks filed on 3/24/05, Applicant submitted a Proposed Amendment and requested for it to be entered by Examiner's Amendment if the arguments made were not accepted. The Proposed Amendment is not entered and claims 1-83 from the Applicant response filed on 9/16/04 will be now considered for examination.

Claim Rejections - 35 USC § 112

The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

3. Claims 52, 79, and 83 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention. The limitation of "the linkage return address being deliberately chosen so that an attempt to execute an instruction from the linkage return address on return from the service routine will raise a program execution exception" (in claim 79) could not be found in the Specification.

The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

4. Claims 1-83 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

a. The following is indefinite:

i. In claim 1, it is unclear in the claim language whether "a pre-existing operating system" (line 2) is the same as "the operating system" (line 3) or if they are different. It is not made clear whether there are one or two operating systems in claim 1.

ii. Claim 56 is rejected for the same indefinite reasons as stated above in the rejection of claim 1.

iii. In claim 5, it is unclear in the claim language whether "a pre-existing thread scheduler" (line 2) is the same as "the thread scheduler" (line 4) or if they are different. It is not made clear whether there are one or two thread schedulers in claim 5.

iv. In claim 33, it is unclear in the claim language whether "a computer operating system" (line 2) is the same as "the operating system complementary.." (line 4) or if they are different. It is not made clear whether there are one or two operating systems in claim 33.

b. The following lacks antecedent basis:

i. Claim 1, "the operating system", line 3, etc.;

ii. Claim 5, "the thread scheduler", line 4, etc.;

iii. Claim 16, "the thread execution mode" and "the thread scheduler execution mode", lines 1-2;

iv. Claim 33, "the operating system complementary to one of the specified entries", line s 4-5;

v. Claim 33, "the original operating system entry", line 8;

vi. Claim 41, "the two operating systems", line 3;

vii. Claim 43, "the process", line 1;

viii. Claim 43, "the context", line 3;

ix. Claim 43, "the process execution mode" and "the operating system execution mode", line 4;

x. Claim 46, "the extended context", lines 2-3;

xi. Claim 47, "the context", line 2;

xii. Claim 56, "the operating system", line 3;

xiii. Claim 72, "the linkage register", line 1;

xiv. Claim 78, "the queue", line 5, etc;

xv. Claim 79, "the service", line 3;

xvi. Claim 82, "modified context", line 5.

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

2. Claims 33, 40-44, 50, 52, and 79 are rejected under 35 U.S.C. 102(e) as being anticipated by Robinson (US 6,199,095 B1).

3. As to claim 33, Robinson teaches a method (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)* comprising:

establishing an entry exception to be raised on each entry to a computer operating system at a specified entry point or on a specified condition (exception handler transfers control to the entry point) *(col. 41, lines 64-67 through col. 42, lines 1-14, col. 33, lines 48-55)*;

establishing a resumption exception to be raised on each resumption from the operation system complementary to one of the specified entries (resuming execution to entry point and a translated routine executes a return instruction to return control to its caller routine) *(col. 31, lines 9-38, col. 26, lines 49-55, col. 33, lines 29-55)*;

on detecting a specified entry to the operating system from an interrupted process of the computer, raising and servicing the entry exception, and then entering the operating system to perform a service associated with the original operating system entry (translated routine entry point and entry to interpreter) *(col. 32, lines 24-44)*; and

on detecting a complementary resumption, raising and servicing the resumption exception, and returning control to the interrupted process (resuming execution to entry point and

a translated routine executes a return instruction to return control to its caller routine) *(col. 31, lines 9-38, col. 26, lines 29-55, col. 32, lines 24-44))*.

4. As to claim 40, Robinson teaches wherein the operating system is an operating system for a computer architecture other than the architecture native to the computer, unmodified for execution on the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)*.

5. As to claim 41, Robinson teaches wherein the computer additionally executes an operating system native to the computer, and each exception is classified for handling by one of the two operating systems (native and non-native) *(col. 29, lines 54-67 through col. 30, lines 1-5, etc.)*.

6. As to claim 42, Robinson teaches wherein operating system and the interrupted thread execute in different instruction set architectures of the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)*.

7. As to claim 43, Robinson teaches wherein the operating system and the process execute in different execution modes of the computer, and the steps to maintain the association between the process and the context are automatically invoked, without explicit software request, on a

transition between the process execution mode and the operating system execution mode (background mode/system 34 and translating, etc.) *(col. 9, lines 29-67)*.

8. As to claim 44, Robinson teaches wherein the process execution mode and the operating system execution mode are two different instruction set architectures of the computer (background mode/system 34 and translating, etc.) *(col. 9, lines 29-67)*.

9. As to claim 50, Robinson teaches further comprising the step of modifying a linkage return address for the process to include information used to maintain the association *(col. 26, lines 49-67, col. 28, lines 13-56, col. 29, lines 18-33)*.

10. As to claim 52, Robninson teaches as part of servicing the entry exception, modifying a linkage return address of the interrupted process, the return address being deliberately chosen so that an attempt to execute an instruction from the return address on return from the operating system will raise the resumption exception *(col. 26, lines 49-67, col. 28, lines 13-56, col. 29, lines 18-33)*.

11. As to claim 79, Robinson teaches a method, comprising:

during invocation of a service routine (translated routine executes a call instruction) of a computer, passing a linkage return address (native dynamic link or "dylink" or linkage) to the service routine at which to resume execution on completion of the service (resume execution in the translated routine after the called routine has completed), the linkage return address being

deliberately chosen (from the shadow stack 212) so that an attempt to execute an instruction from the linkage return address on return from the service routine will raise a program execution exception *(col. 26, lines 49-67, col. 28, lines 13-56, col. 29, lines 18-33)*;

on return from the service routine, attempting to execute the instruction at the linkage return address ("dylnk", etc.) and raising the chosen exception *(routine return address that is pushed onto the stack by the program when it executes a call instruction, etc.) (col. 26, lines 1-13 and 49-67, col. 49, lines 1-10)*; and

after servicing the exception, returning control to a caller of the service routine (resuming execution to entry point and a translated routine executes a return instruction to return control to its caller routine) *(col. 31, lines 9-38, col. 26, lines 49-55, col. 33, lines 29-55)*.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

12. Claims 5-13, 15-18, 20-24, 26, 30-, and 82-83 are rejected under 35 U.S.C. 103(a) as being unpatentable over Robinson (US 6,199,095 B1) in view of Bitar et al. (hereinafter Bitar) (US 6,418,460 B1).

13. As to claim 5, Robinson teaches a method (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)*, comprising:

managing and excuting concurrent threads of control, each thread having an associated context (non-native return stack 211 or a single register CONTEXT 105), an association between a thread and a set of computer resources of the associated context (thread associated context data structure, which also includes copies of integer registers) *(col. 25, lines 11-52, col. 16, lines 1-9)*; and

without modifying the thread scheduler to, maintaining an association between one of the threads and an extended context (shadow stack 212 and linked list of context data structures or an additional register) *(col. 25, lines 11-52, col. 16, lines 1-9)* of the thread through a context change (context switch or to change from non-native instructions into native instructions, etc.) *(col. 27, lines 19-26)*, the extended context including resources (registers, etc.) of the computer associated with the thread that are beyond those resources (additional registers, etc.) whose association with the thread.

Robinson teaches a process scheduler as a software service process which, amongst other things, is used to schedule various transactions (such as the translations, etc.) within and between the run-time and background systems *(col. 10, lines 25-56)*. However, Robinson fails to explicitly state having a thread scheduler. Bitar teaches a thread scheduler (kernel-level scheduler 28 or User-level scheduler 26 in Fig. 3) and context switching for a multithreaded environment *(col. 12, lines 26-45, col. 13, lines 3-17, col. 7, lines 45-67 through col. 8, lines 1-8, see claim 19)*. It would have been obvious to one of ordinary skill in the art at the time the

invention was made to combine the references of Robinson and Bitar because it would increase control and will result in better scheduling decisions *(col. 5, lines 22-27)*.

14. As to claim 6, Robinson teaches wherein the thread scheduler is a component of an operating system of the computer, and further comprising:

establishing an entry exception to be raised on each entry to the operating system at a specified entry point or on a specified condition (exception handler transfers control to the entry point) *(col. 41, lines 64-67 through col. 42, lines 1-14, col. 33, lines 48-55)*;

establishing a resumption exception to be raised on a resumption from the operating system following on a specified entry (resuming execution to entry point and a translated routine executes a return instruction to return control to its caller routine) *(col. 31, lines 9-38, col. 26, lines 49-55, col. 33, lines 29-55)*;

on detecting a specified entry to the operating system from an interrupted process of the computer, raising the entry exception, and establishing the association as part of servicing the entry exception (translated routine entry point and entry to interpreter) *(col. 32, lines 24-44);* and

raising the resumption exception, and as part of servicing the resumption exception, reestablishing the context in association with the resumed thread returning control to the interrupted process (resuming execution to entry point and a translated routine executes a return instruction to return control to its caller routine) *(col. 31, lines 9-38, col. 26, lines 29-55, col. 32, lines 24-44))*.

15. As to claim 7, Robinson teaches wherein an exception handler for the entry exception is programmed to save a context of the interrupted process and modify the thread context before delivering the modified context to the operating system; and an exception handler for the resumption exception is programmed to restore the context saved by a corresponding execution of the entry exception handler (exception handler transfers control to the entry point) *(col. 41, lines 64-67 through col. 42, lines 1-14, col. 33, lines 48-55, see Abstract, col. 26, liens 33-67).*

16. As to claim 8, Robinson teaches wherein the operating system is an operating system for a computer architecture other than the architecture native to the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25).*

17. As to claim 9, Robinson teaches wherein the computer additionally executes an operating system native to the computer, and each exception is classified for handling by one of the two operating systems *(col. 29, lines 54-67 through col. 30, lines 1-5, etc.).*

18. As to claim 10, Robinson teaches wherein operating system and the interrupted thread execute in different instruction set architectures of the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25, col. 29, lines 54-67 through col. 30, lines 1-5, etc.).*

19. As to claim 11, Robinson teaches wherein the operating system is in a binary code for a computer architecture non-native to the architecture of the computer (binary image conversion system converts instructions from a instruction set of a first, non native computer system to a second, different native computer system) *(see Abstract)*.

20. As to claim 12, Robinson teaches wherein the computer additionally executes an operating system native to the computer, and each exception is classified for handling by one of the two operating systems *(col. 29, lines 54-67 through col. 30, lines 1-5, etc.)*.

21. As to claim 13, Robinson teaches wherein operating system and the interrupted thread execute in different instruction set architectures of the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25, col. 29, lines 54-67 through col. 30, lines 1-5, etc.)*.

22. As to claim 15, Robinson teaches wherein thread scheduler and the thread execute in different execution modes of the computer, and the steps to maintain the association between the thread and the context are automatically invoked, without explicit software request, on a transition between the thread execution mode and the thread scheduler execution mode (background mode/system 34 and translating, etc.) *(col. 9, lines 29-67, col. 2, lines 6-25, col. 41, lines 64-67 through col. 42, lines 1-14, col. 33, lines 48-55)*.

23. As to claim 16, Robinson teaches wherein the thread execution mode and the thread scheduler execution mode are two different instruction set architectures of the computer (background mode/system 34 and translating, etc.) *(col. 9, lines 29-67, col. 2, lines 6-25, col. 29, lines 54-67 through col. 30, lines 1-5, etc.)*.

24. As to claim 17, Robinson teaches during servicing the entry exception, saving a portion of the context of the computer, and altering the context of an interrupted thread before delivering the interrupted thread and its corresponding context to the operating system *(see Abstract, col. 26, liens 33-67)*.

25. As to claim 18, Robinson teaches the step of modifying a linkage return address for resumption of the thread to include information used to maintain the association *(col. 26, lines 49-67, col. 28, lines 13-56, col. 29, lines 18-33)*.

26. As to claim 20, Robinson (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)* in view of Bitar *(col. 12, lines 26-45, col. 13, lines 3-17, col. 7, lines 45-67 through col. 8, lines 1-8, see claim 19)* teaches wherein the thread scheduler is an operating system for a computer architecture other than the architecture native to the computer.

27. As to claim 21, Robinson teaches wherein the computer additionally executes an operating system native to the computer, and each exception is classified for handling by one of the two operating systems *(col. 29, lines 54-67 through col. 30, lines 1-5, etc.)*.

28. As to claim 22, Robinson teaches wherein operating system and the interrupted thread execute in different instruction set architectures of the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25, col. 29, lines 54-67 through col. 30, lines 1-5, etc.)*.

29. As to claim 23, Robinson teaches wherein thread scheduler and the thread execute in different execution modes of the computer, and the steps to maintain the association between the thread and the context are automatically invoked, without explicit software request, on a transition between the thread execution mode and the thread scheduler execution mode (background mode/system 34 and translating, etc.) *(col. 9, lines 29-67)*.

30. As to claim 24, Robinson teaches wherein the thread execution mode and the thread scheduler execution mode are two different instruction set architectures of the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25, col. 9, lines 29-67)*.

31. As to claim 26, Robinson teaches: in an interrupt handler of the computer, saving a portion of the context of the computer, and altering the context of an interrupted thread before delivering the interrupted thread and its corresponding context to the thread scheduler *(see Abstract, col. 26, liens 33-67)*.

32. As to claim 30, Robinson teaches modifying a linkage return address for the thread to include information used to maintain the association *(col. 26, lines 49-67, col. 28, lines 13-56, col. 29, lines 18-33)*.

33. As to claim 82, Robinson teaches without modifying a pre-existing thread scheduler of the computer, establishing an entry handler (background system 34 or the interpreter 44, or part of the exception handler, etc.) for execution at a specified entry point or on a specified entry condition to the thread scheduler (entry to second architecture from first architecture, etc.), the entry handler programmed to save a context of an interrupted thread and modify the thread context before delivering the modified context to the thread scheduler (snap-shot of the current state saved in context data structure) and modify the thread context before delivering the modified context to the operating system (translate from non-native to native or temporary storage for logical register manipulations, e.g.) *(col. 25, lines 10-32, col. 11, lines 10-20, col. 27, lines 1-15)*.

34. As to claim 83, Robinson teaches a method, comprising:

during invocation of a service routine (translated routine executes a call instruction) of a computer, passing a linkage return address (native dynamic link or "dylink" or linkage) to the service routine at which to resume execution on completion of the service (resume execution in the translated routine after the called routine has completed), the linkage return address being deliberately chosen (from the shadow stack 212) so that an attempt to execute an instruction from the linkage return address on return from the service routine will raise a program execution exception *(col. 26, lines 49-67, col. 28, lines 13-56, col. 29, lines 18-33)*;

on return from the service routine, attempting to execute the instruction at the linkage return address ("dylnk", etc.) and raising the chosen exception *(routine return address that is pushed onto the stack by the program when it executes a call instruction, etc.) (col. 26, lines 1-13 and 49-67, col. 49, lines 1-10)*; and

after servicing the exception, returning control to a caller of the service routine (resuming execution to entry point and a translated routine executes a return instruction to return control to its caller routine) *(col. 31, lines 9-38, col. 26, lines 49-55, col. 33, lines 29-55)*.

35. **Claims 19, 27 and 31 are rejected under 35 U.S.C. 103(a) as being unpatentable over Robinson (US 6,199,095 B1) in view of Bitar et al. (hereinafter Bitar) (US 6,418,460 B1), and further in view of Mann (US 6,154,857).**

36. As to claim 19, Robinson fails to explicitly teach wherein the modification leaves at least half of the bits of the linkage return address intact. However, Mann teaches modifying/updating

at least half (one of the halves or both halves) of the data registers (col. 11, lines 10-13). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of modifying/updating at least half (one of the halves or both halves) of the data registers to the registers of Robinson because efficiency is increased by not modifying any halves that are unnecessary *(col. 11, lines 10-13)*.

37. As to claim 27, Robinson teaches wherein the operating-system-maintained resources of the thread context include data registers of the non-native computer architecture (translating between native and non-native using registers) *(see rejection of claim 5)*. Robinson fails to explicitly teach modifying at least half of the data registers. However, Mann teaches modifying/updating at least half (one of the halves or both halves) of the data registers (col. 11, lines 10-13). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of modifying/updating at least half (one of the halves or both halves) of the data registers to the registers of Robinson because efficiency is increased by not modifying any halves that are unnecessary *(col. 11, lines 10-13)*.

38. As to claim 31, Robinson fails to explicitly teach modifying at least half of the data registers. However, Mann teaches modifying/updating at least half (one of the halves or both halves) of the data registers (col. 11, lines 10-13). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of modifying/updating at least half (one of the halves or both halves) of the data registers to the registers of Robinson

because efficiency is increased by not modifying any halves that are unnecessary *(col. 11, lines 10-13)*.

39. **Claims 34-36, 38-39, 56-60, 62-67, 71-72, and 74-75 are rejected under 35 U.S.C. 103(a) as being unpatentable over Robinson (US 6,199,095 B1) in view of Fleck et al. (hereinafter Fleck) (US 6,128,641).**

40. As to claim 34, Robinson *(col. 25, lines 10-32, col. 11, lines 10-20, col. 27, lines 1-15)* in view of Fleck *(col. 1, lines 47-55, col. 2, lines 7-31)* teaches wherein an exception handler for the entry exception is programmed to save a context of the interrupted process and modify the thread context before delivering the modified context to the operating system; and an exception handler for the resumption exception is programmed. Robinson does not explicitly state to restore the context that is saved. However, Fleck teaches context switching which includes the following: context save areas *(col. 1, lines 47-55)*; saving contexts resulting from a sequence of calls, traps, or interrupts; exiting a called function or trap or interrupt handler and switching back to the previous context *(col. 2, lines 7-10 and 24-25)*; instructions to return from an interrupt or trap handler *(col. 2, lines 15-18)*; restoring the saved context (col. 2, lines 19-45); and resuming execution after exiting *(col. 2, lines 30-31)*. It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the references of Robinson and Fleck because it would reduce memory space and execution overhead for both task switching and for function calls and returns *(col. 1, lines 35-44)*.

41. As to claim 35, Robinson teaches wherein the operating system is an operating system for a computer architecture other than the architecture native to the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)*.

42. As to claim 36, Robinson teaches wherein operating system and the interrupted thread execute in different instruction set architectures of the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)*.

43. As to claim 38, Robinson teaches wherein the operating system and the process execute in two different instruction set architectures of the computer, and at least some of the steps to maintain the association between the process and the context are automatically invoked, without explicit software request, on a transition between the instruction set architectures *(col. 2, lines 6-25, col. 41, lines 64-67 through col. 42, lines 1-14, col. 33, lines 48-55)*.

44. As to claim 39, Robinson *(col. 26, lines 49-67, col. 28, lines 13-56, col. 29, lines 18-33)* in view of Fleck *(col. 1, lines 47-55, col. 2, lines 7-31)* teaches modifying a linkage return address for the process to include information used to restore the context.

45. As to claim 56, Robinson teaches a method (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)*, comprising:

without modifying a pre-existing operating system of the computer, establishing an entry handler (background system 34 or the interpreter 44, or part of the exception handler, etc.) for execution at a specified entry point or on a specified entry condition to the operating system (entry to second architecture from first architecture, etc.), the entry handler programmed to save a context of an interrupted thread (snap-shot of the current state saved in context data structure) and modify the thread context before delivering the modified context to the operating system (translate from non-native to native or temporary storage for logical register manipulations, e.g.) *(col. 25, lines 10-32, col. 11, lines 10-20, col. 27, lines 1-15)*;

without modifying the operating system, establishing an exit handler (part of the exception handler, etc.) for execution on resumption (resuming execution to entry point and a translated routine executes a return instruction to return control to its caller routine) from the operating system following an entry through the entry handler (part of the exception handler) *(col. 31, lines 9-38, col. 26, lines 49-67, col. 33, lines 29-55)*.

Robinson does not explicitly state to restore the context that is saved. However, Fleck teaches context switching which includes the following: context save areas *(col. 1, lines 47-55)*; saving contexts resulting from a sequence of calls, traps, or interrupts; exiting a called function or trap or interrupt handler and switching back to the previous context *(col. 2, lines 7-10 and 24-25)*; instructions to return from an interrupt or trap handler *(col. 2, lines 15-18)*; restoring the saved context (col. 2, lines 19-45); and resuming execution after exiting *(col. 2, lines 30-31)*. It

would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the references of Robinson and Fleck because it would reduce memory space and execution overhead for both task switching and for function calls and returns *(col. 1, lines 35-44)*.

46. As to claim 57, Robinson teaches:

scheduling concurrent threads of control by the operating system, each thread having an associated context (non-native return stack 211 or a single register CONTEXT 105), an association between a thread and a set of computer resources of the associated context being maintained by the operating system (thread associated context data structure, which also includes copies of integer registers) *(col. 25, lines 11-52, col. 16, lines 1-9)*; and

the entry and exit handlers (part of the exception handler, etc.) being programmed to maintain an association between one of the threads and an extended context of the thread through a context change induced by the operating system, the extended context including resources of the computer associated with the thread that are beyond those resources (additional registers, etc.) whose association with the thread is maintained by the operating system (context switch or to change from non-native instructions into native instructions, etc.) *(col. 27, lines 19-26)*.

47. As to claim 58, Robinson teaches wherein the operating system is an operating system for a computer architecture other than the architecture native to the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)*.

48. As to claim 59, Robinson teaches wherein the operating system and the thread execute in different execution modes of the computer, and the steps to maintain the association between the thread and the context are automatically invoked, without explicit software request, on a transition between the thread execution mode and the operating system execution mode (background mode/system 34 and translating, etc.) *(col. 9, lines 29-67)*.

49. As to claim 60, Robinson teaches in the entry handler, saving a portion of the context of the computer, and altering the context of the interrupted thread before delivering the interrupted thread and its corresponding context to the operating system *(see Abstract, col. 26, liens 33-67)*.

50. As to claim 62, Robinson teaches modifying a linkage return address for the thread to include information used to maintain the association *(col. 26, lines 49-67, col. 28, lines 13-56, col. 29, lines 18-33)*.

51. As to claim 63, Robinson teaches wherein the operating system is an operating system for a computer architecture other than the architecture native to the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)*.

52. As to claim 64, Robinson teaches wherein the computer additionally executes an operating system native to the computer, and each interrupt or exception is classified for

handling by one of the two operating systems *(col. 29, lines 54-67 through col. 30, lines 1-5, etc.)*.

53. As to claim 65, Robinson teaches wherein operating system and the interrupted thread execute in different instruction set architectures of the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)*.

54. As to claim 66, Robinson teaches wherein the operating system and the thread execute in different execution modes of the computer, and the steps to maintain the association between the thread and the context are automatically invoked, without explicit software request, on a transition between the thread execution mode and the operating system execution mode (background mode/system 34 and translating, etc.) *(col. 9, lines 29-67)*.

55. As to claim 67, Robinson teaches wherein the thread execution mode and the operating system execution mode are two different instruction set architectures of the computer (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25, col. 9, lines 29-67)*.

56. As to claim 71, Robinson *(col. 26, lines 49-67, col. 28, lines 13-56, col. 29, lines 18-33)* in view of Fleck *(col. 1, lines 47-55, col. 2, lines 7-31)* teaches modifying a linkage return address for the thread to include information used to restore the context of the thread.

57. As to claim 72, Robinson teaches wherein the linkage register is modified with information indicating an execution path by which, or a condition on which, execution arrived at the entry handler *(col. 26, lines 49-67, col. 28, lines 13-56, col. 29, lines 18-33)*.

58. As to claim 74, Robinson teaches wherein the linkage register is modified with information indicating a storage location at which at least the portion of the thread context to be modified is saved before the modifying *(col. 26, lines 49-67, col. 28, lines 13-56, col. 29, lines 18-33)*.

59. As to claim 75, Robinson teaches:

wherein the interrupted thread at the point of interruption executes in one instruction set architecture and the operating system is coded primarily in a different instruction set architecture (of executing on a first architecture or native code but accessed from a client executing in an execution engine of a second architecture or non-native code) *(col. 2, lines 6-25)*; and

further comprising the step of setting of a register to a value that specifies actions to be taken by the entry handler or exit handler to convert operands from one form to another to conform to a data storage convention of the operating system instruction set architecture (exception handler transfers control to the entry point or translate from non-native to native or

temporary storage for logical register manipulations, e.g.) *(col. 25, lines 10-32, col. 11, lines 10-20, col. 27, lines 1-15, col. 41, lines 64-67 through col. 42, lines 1-14, col. 33, lines 48-55)*.

60. **Claims 45-46 are rejected under 35 U.S.C. 103(a) as being unpatentable over Robinson (US 6,199,095 B1) in view of Mann (US 6,154,857).**

61. As to claim 45, Robinson teaches a service routine for the entry exception and modifying data registers in association with the process by the operating system before delivering the process to the non-native operating system (see rejection of claim 33). Robinson fails to explicitly teach modifying at least half of the data registers. However, Mann teaches modifying/updating at least half (one of the halves or both halves) of the data registers (col. 11, lines 10-13). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of modifying/updating at least half (one of the halves or both halves) of the data registers to the registers of Robinson because efficiency is increased by not modifying any halves that are unnecessary *(col. 11, lines 10-13)*.

62. As to claim 46, Robinson teaches wherein at least some of the modified registers are overwritten by information indicating a storage location at which at least the extended context, the extended context being the resources beyond those whose resource association with the process is maintained by the operating system, is saved before the modifying *(col. 25, lines 10-32, col. 11, lines 10-20, col. 27, lines 1-15)*.

63. **Claims 61, 68-70, and 73 are rejected under 35 U.S.C. 103(a) as being unpatentable over Robinson (US 6,199,095 B1) in view of Fleck et al. (hereinafter Fleck) (US 6,128,641), and further in view of Mann (US 6,154,857).**

64. As to claims 61, 68, and 73, Robinson teaches a service routine for the entry exception and modifying data registers in association with the process by the operating system before delivering the process to the non-native operating system (see rejection of claim 33). Robinson fails to explicitly teach modifying at least half of the data registers. However, Mann teaches modifying/updating at least half (one of the halves or both halves) of the data registers (col. 11, lines 10-13). It would have been obvious to one of ordinary skill in the art at the time the invention was made to include the feature of modifying/updating at least half (one of the halves or both halves) of the data registers to the registers of Robinson because efficiency is increased by not modifying any halves that are unnecessary *(col. 11, lines 10-13)*.

65. As to claim 69, Robinson teaches wherein at least some of the modified registers are overwritten by information indicating a storage location at which at least the portion of the thread context to be modified is saved before the modifying *(col. 25, lines 10-52, col. 11, lines 10-20, col. 27, lines 1-15)*.

66. As to claim 70, Robinson teaches wherein at least some of the modified registers are overwritten by a value that enables validation of the contents of the context *(col. 25, lines 10-52, col. 11, lines 10-20, col. 27, lines 1-15)*.

Allowable Subject Matter

67. The indicated allowable subject matter (given in Applicant Interview on 9/7/05) in claims 6-13, 15-19, 46, 52, and 83 are withdrawn in view of the newly discovered reference(s) to US 6,199,095 B1. Rejections based on the newly cited reference(s) are above.

68. Claims 1-4 would be allowable if rewritten or amended to overcome the rejection(s) under 35 U.S.C. 112, 2nd paragraph, set forth in this Office action.

69. Claims 14, 25, 28-29, 32, 37, 47-49, 51, 53-55, 76-78, and 80-81 would be allowable if rewritten to overcome the rejection(s) under 35 U.S.C. 112, 2nd paragraph, set forth in this Office action and to include all of the limitations of the base claim and any intervening claims.

Response to Arguments

70. Applicant's arguments have been fully considered but are moot in view of the new grounds of rejections.

Conclusion

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure:

Appell et al. (US 4,077,058) teaches a context (process control block includes information which specifies the state of a processor) associated with an extended context (associated with each process control block is a décor extension table having information to indicate whether a specified function, such as the emulation of another processor, may be executed in the system) to translate between native and non-native instruction sets *(see Abstract)*.

Yates et al. (US 5,930,509) teaches a method and apparatus for performing binary translation that converts instructions from a instruction set of a first, non native computer system to a second, different, native computer system *(see Abstract)*.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kenneth Tang whose telephone number is (571) 272-3772. The examiner can normally be reached on 8:30AM - 6:00PM, Every other Friday off.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on (571) 272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Kt
9/17/05

MENG-AL T. AN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100